SUGHRUE MION, PLLC

2100 Pennsylvania Avenue, NW
Washington, DC 20037-3213
T 202.293.7060
F 202.293.7860

www.sughrue.com

FAX

Date	May 7, 2009
To	Examiner Angela Lie
Of	United Staes Patent and Trademark Office
Fax	571-273-8445
From	Ruthleen E. Uy (619) 238-3545
Subject	Application No.: 10/802,471
Our Ref	Q99144
Pages	7

(including cover sheet)

Please call attention to problems with this transmission by return fax or telephone. Thank you.

TO: Examiner Angela Lie (Fax: 571-273-8445)
FROM: Ruthleen Uy (619-238-3545)
Re: Application No.: 10/802,471; Q99144

LISTING OF CLAIMS:

1. (previously presented): A method for processing a multiplicity of data update requests made by a customer, said method comprising the steps of:

grouping all of said data update requests which is followed by updating of the corresponding data into a predetermined plurality of blocks for execution by a data processor, the data update requests within each of said blocks and from one of said blocks to a next one of said blocks being arranged in an order that said data update requests need to be executed to yield a proper data result, each of said blocks having approximately a same capacity for said data update requests, said capacity corresponding to a number of said data update requests which said data processor is adapted to efficiently process in order before processing said data update requests in the next one of said blocks; and

then said data processor processing said data update requests within said one block in said order, and then said data processor processing said data update requests within said next block in said order;

wherein the processing the multiplicity of data update requests is performed only for the data update requests.

2. (original): A method as set forth in claim 1 wherein said order is an order in which said data update requests were made.

3. (previously presented): A method as set forth in claim 1 wherein:

said capacity corresponds to a number of said data update requests which said data processing unit is adapted to optimally process in order in said one block before processing said data update requests in said next one of said blocks.

4. (original): A method as set forth in claim 1 wherein said data update requests within each of said blocks are arranged into said order by order information stored within or associated with said blocks.

5. (previously presented): A system for processing a multiplicity of data update requests made by a customer, said system comprising:

means for grouping all of said data update requests stored in a database which is followed by updating of the corresponding data into a predetermined plurality of blocks for execution by a data processor, the data update requests within each of said blocks and from one of said blocks to a next one of said blocks being arranged in an order that said data update requests need to be executed to yield a proper data result, each of said blocks having approximately a same capacity for said data update requests, said capacity corresponding to a number of said data update requests which said data processor is adapted to efficiently process in order before processing said data update requests in the next one of said blocks;

said data processor including means for then processing said data update requests within said one block in said order;

said data processor including means for then processing said data update requests within said next block in said order;

wherein the processing the multiplicity of data update requests is performed only for the data update requests.

6. (original): A system as set forth in claim 5 wherein said order is an order in which said data update requests were made.

7. (previously presented): A system as set forth in claim 5 wherein:

said capacity corresponds to a number of said data update requests which said data processing unit is adapted to optimally process in order in said one block before processing said data update requests in said next one of said blocks.

8. (previously presented): A computer program product stored on a computer readable medium for processing a multiplicity of data update requests made by a customer, said computer program product comprising:

first program instructions to group all of said data update requests which is followed by updating of the corresponding data into a predetermined plurality of blocks for execution by a data processor, the data update requests within each of said blocks and from one of said blocks to a next one of said blocks being arranged in an order that said data update requests need to be executed to yield a proper data result, each of said blocks having approximately a same capacity for said data update requests, said capacity corresponding to a number of said data update requests which said data processor is adapted to efficiently process in order before processing said data update requests in the next one of said blocks; and

second program instructions within said data processor to then process said data update requests within said one block in said order, and third program instructions within said data processor to then process said data update requests within said next block in said order; and wherein

said first, second and third program instructions are recorded on said medium;

wherein the processing the multiplicity of data update requests is performed only for the data update requests.

9. (original): A computer program product as set forth in claim 8 wherein said order is an order in which said data update requests were made.

10. (previously presented): A computer program product as set forth in claim 8 wherein:

said capacity corresponds to a number of said data update requests which said data processing unit is adapted to optimally process in order in said one block before processing said data update requests in said next one of said blocks.

11. (currently amended): A method for processing a multiplicity of first data update requests made by a first customer and a multiplicity of second data update requests made by a second customer, said method comprising the steps of:

grouping all of said first data update requests which is followed by updating of the corresponding data into a predetermined plurality of first blocks for execution by a first data processor unit, the first data update requests within each of said first blocks and from one of said first blocks to a next one of said first blocks being arranged in a first order that said first data update requests need to be executed to yield a proper data result, each of said first blocks having approximately a same first capacity for said first data update requests, said first capacity corresponding to a number of said first data update requests which said first data processing unit is adapted to efficiently process in order before processing said first data update requests in the next one of said first blocks, and then said first data processing unit processing said first data update requests within said one first block in said first order, and then said first data processing unit processing said first data update requests within said next first block in said first order; and

after performing grouping of all of said first data update requests, grouping all of said second data update requests into a plurality of second blocks for execution by a second data processor unit, the second data update requests within each of said second blocks and from one of said second blocks to a next one of said second blocks being arranged in a first order that said second data update requests need to be executed to yield a proper data result, each of said second blocks having approximately a same second capacity for said second data update requests, said second capacity corresponding to a number of said second data update requests which said second data processing unit is adapted to efficiently process in order before processing said second data update requests in the next one of said second blocks, and then said second data processing unit processing said second data update requests within said one second block in said second order, and then said second data processing unit processing said second data update requests within said next second block in said second order,

wherein the processing the multiplicity of data update requests is performed only for the data update requests.

12. (original): A method as set forth in claim 11 wherein said first order is an order in which said first data update requests were made, and said second order is an order in which said second data update requests were made.

13. (previously presented): A method as set forth in claim 11 wherein:

said first capacity corresponds to a number of said first data update requests which said first data processing unit is adapted to optimally process in order in said one first block before processing said first data update requests in the next one of said first blocks; and

said second capacity corresponds to a number of said second data update requests which said second data processing unit is adapted to optimally process in order in said one second block before processing said second data update requests in the next one of said second blocks.

14. (original): A method as set forth in claim 11 wherein said first data processing unit processes said first data update requests in parallel with said second data processing unit processing said second data update requests.

15. (canceled).

16. (canceled).

17. (previously presented): The method according to claim 1, wherein the grouping of all of said data update requests into the plurality of blocks is performed at a same time.

18. (previously presented): The method according to claim 1, wherein said blocks contain a predetermined number of the data update requests.

19. (previously presented): The method according to claim 1, wherein said blocks are grouped into a package according to a common key among the blocks.